In [1]:
```
#5-09-23_Tupakula_Vaishnavi
```

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
data=pd.read_csv("Titanic-Dataset.csv")
```

In [3]:
```
data.head()
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [4]:
```
data.tail()
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [6]:
```
data.describe()
```

Out[6]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Handling Null Values

In [7]:
```
data.isnull().any()
```

Out[7]:
```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age             True
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin           True
Embarked        True
dtype: bool
```

In [8]:
```
data.isnull().sum()
```

Out[8]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [9]:
```
mean=data["Age"].mean()
```

Filling the null values in Age column with Mean

```
In [10]: data["Age"]=data["Age"].fillna(mean)
```

```
In [11]: data["Age"].tail()
```

```
Out[11]: 886    27.000000
         887    19.000000
         888    29.699118
         889    26.000000
         890    32.000000
         Name: Age, dtype: float64
```

```
In [12]: data["Age"].isnull().sum()
```

```
Out[12]: 0
```

Filling the Null values in Embarked with mode

```
In [13]: Em_mode=data["Embarked"].mode()
```

```
In [14]: data["Embarked"]=data["Embarked"].fillna(Em_mode[0])
```

```
In [15]: data["Embarked"].isnull().sum()
```

```
Out[15]: 0
```

```
In [ ]:
```

Filling the null values in Cabin with mode

```
In [16]: Cabin_mode=data["Cabin"].mode()
```

```
In [17]: data["Cabin"]
```

```
Out[17]: 0        NaN
         1        C85
         2        NaN
         3       C123
         4        NaN
                ...
         886      NaN
         887      B42
         888      NaN
         889     C148
         890      NaN
         Name: Cabin, Length: 891, dtype: object
```

```
In [18]: Cabin_mode
```

```
Out[18]: 0        B96 B98
         1    C23 C25 C27
         2             G6
         Name: Cabin, dtype: object
```

```
In [19]: data["Cabin"]=data["Cabin"].fillna(Cabin_mode[2])
```

```
In [20]: data["Cabin"].isnull().sum()
```

```
Out[20]: 0
```

In [21]:

```
data["Cabin"]
```

Out[21]:

```
0        G6
1       C85
2        G6
3      C123
4        G6
       ...
886      G6
887     B42
888      G6
889    C148
890      G6
Name: Cabin, Length: 891, dtype: object
```

In [22]:

```
data.isnull().sum()
```

Out[22]:

```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
```

Data Visualisation

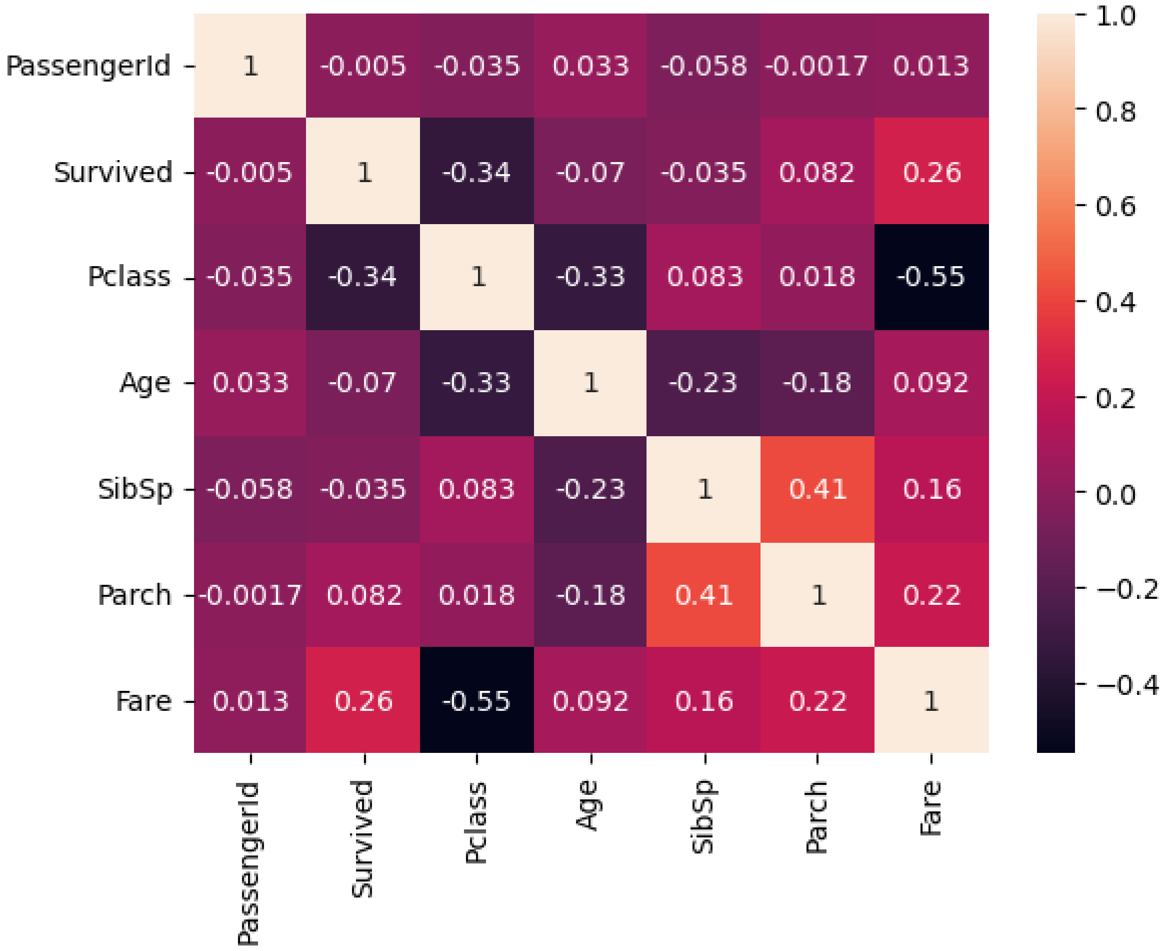
In [23]:

```
cor=data.corr()
```

```
C:\Users\pichi\AppData\Local\Temp\ipykernel_20180\1426905697.py:1: FutureWarning: The default value of numeric_only in DataFra
me.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeri
c_only to silence this warning.
  cor=data.corr()
```

```
In [24]: sns.heatmap(cor,annot=True)
```

```
Out[24]: <Axes: >
```



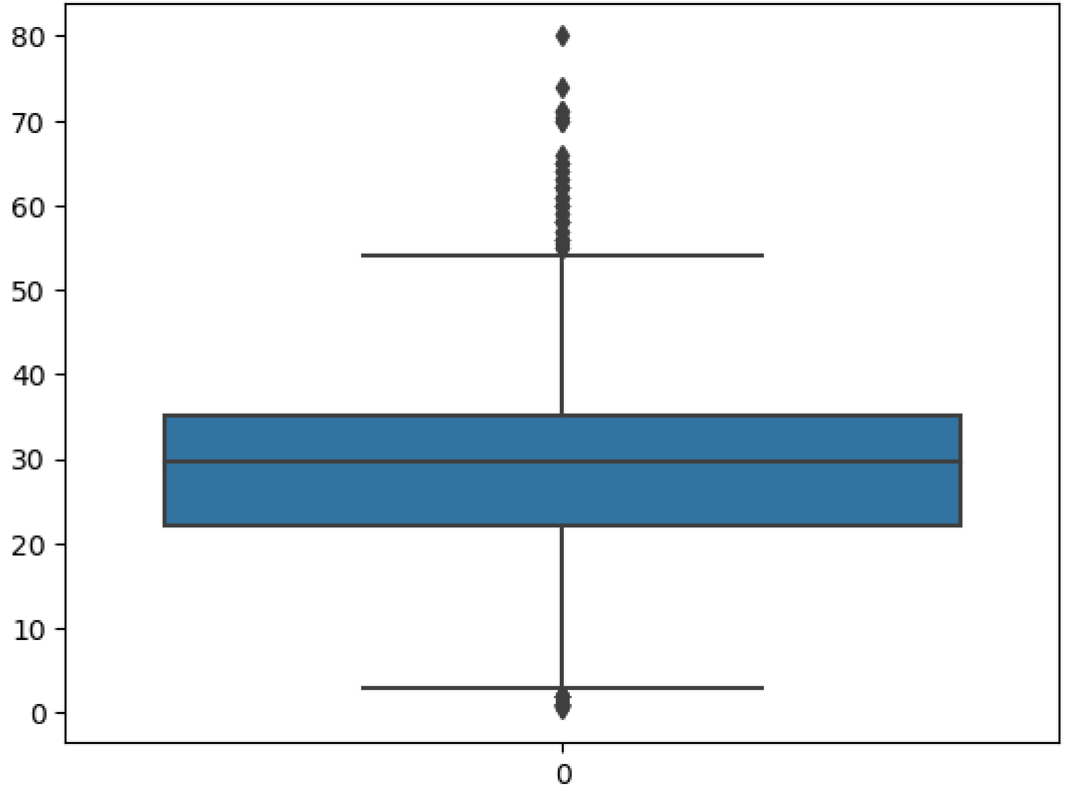
Handling the outliers

In [25]:

```
sns.boxplot(data["Age"])
```

Out[25]:

```
<Axes: >
```



Outliers

In [26]:

```
Age_q1 = data.Age.quantile(0.25)
Age_q3 = data.Age.quantile(0.75)
print(Age_q1)
print(Age_q3)
```

```
22.0
35.0
```

In [27]:

```
IQR_Age=Age_q3-Age_q1
IQR_Age
```

Out[27]:

```
13.0
```

In [28]:

```
upperlimit_Age=Age_q3+1.5*IQR_Age
upperlimit_Age
```

Out[28]:

```
54.5
```

In [29]:

```
lower_limit_Age = Age_q1-1.5*IQR_Age
lower_limit_Age
```

Out[29]:

```
2.5
```

In [30]:

```
median_Age=data["Age"].median()
median_Age
```

Out[30]:

```
29.69911764705882
```

In [32]:

```
data["Age"]=np.where(data["Age"]>upperlimit_Age,median_Age,data["Age"])
```

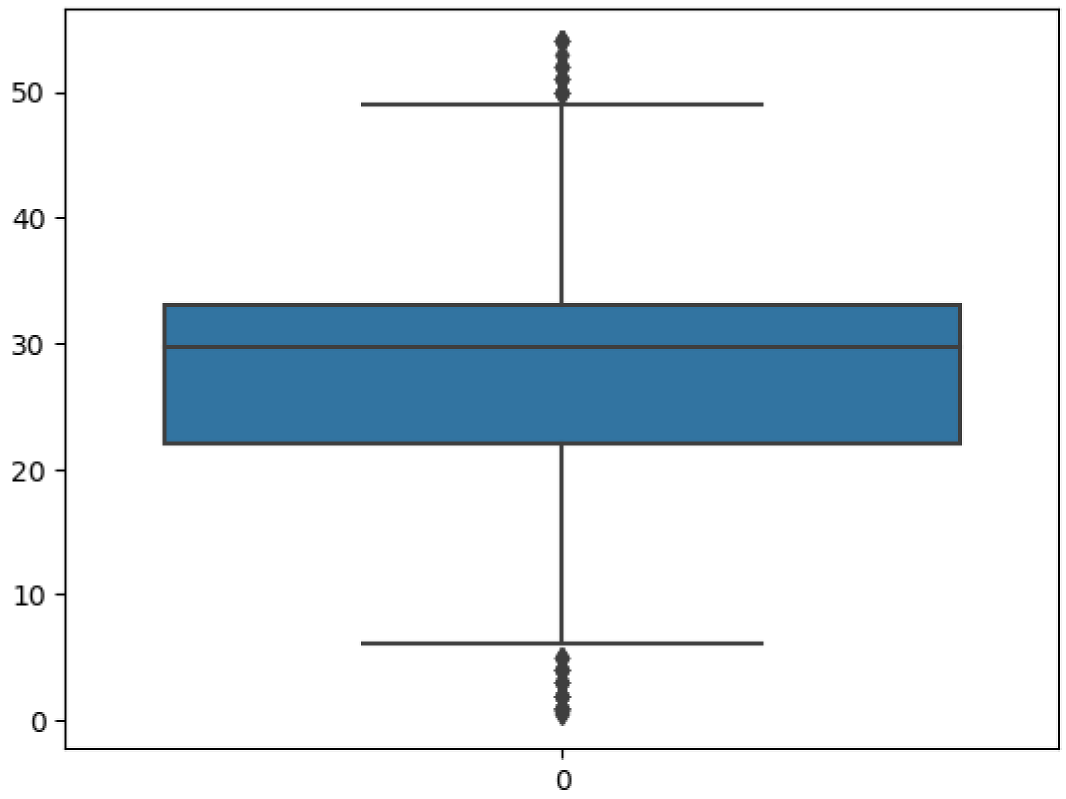
In [33]:

```
(data["Age"]>54.5).sum()
```

Out[33]:

```
0
```

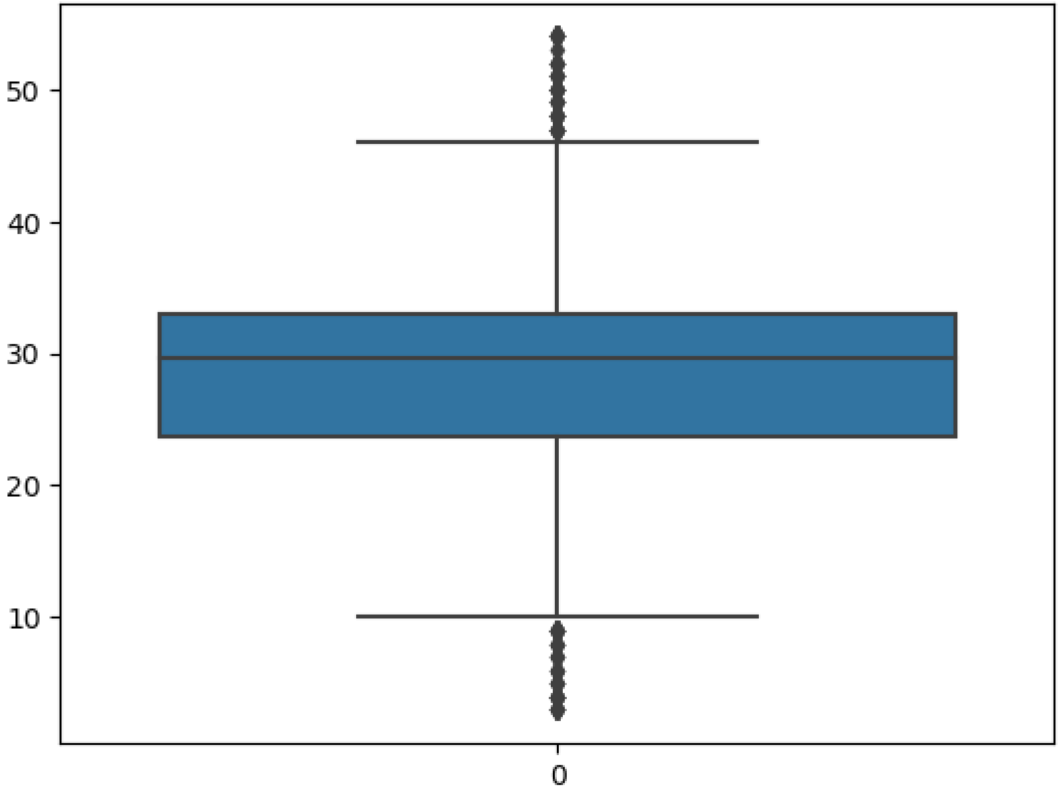
```
In [34]: sns.boxplot(data["Age"])

Out[34]: <Axes: >
```



```
In [35]: data["Age"]=np.where(data["Age"]<lower_limit_Age,median_Age,data["Age"])
```

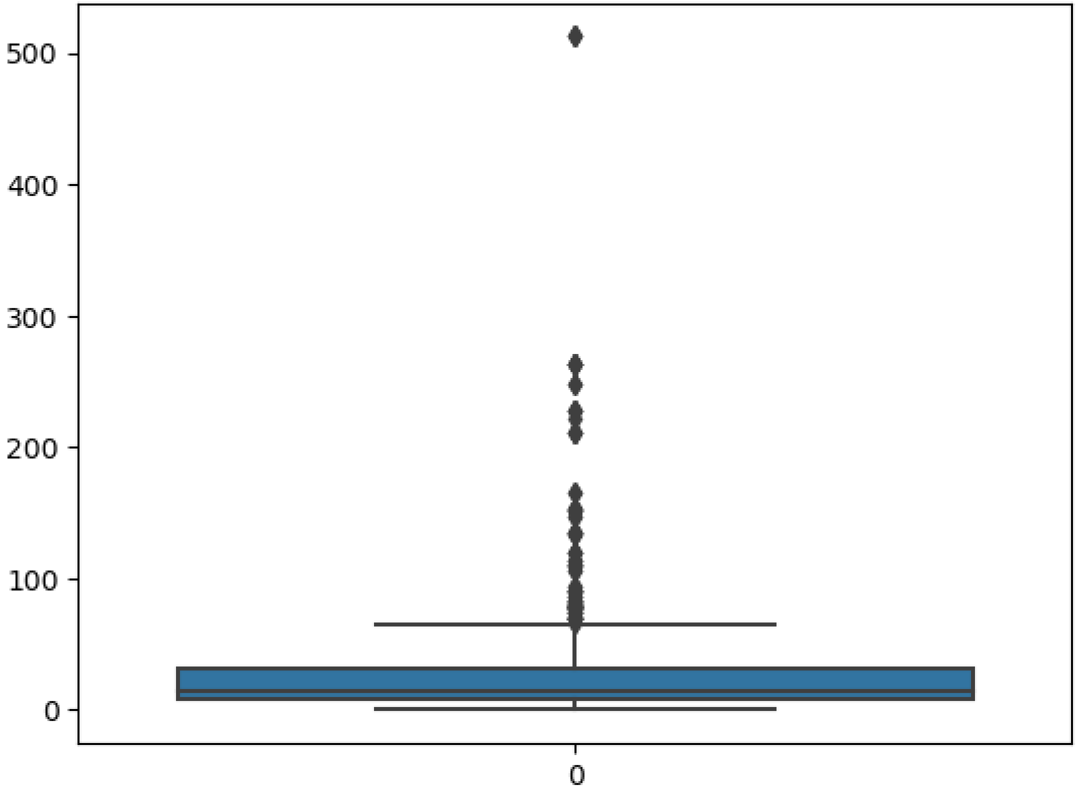
In [36]:
```
sns.boxplot(data["Age"])
```

Out[36]:
```
<Axes: >
```



```
In [37]: sns.boxplot(data["Fare"])
```

```
Out[37]: <Axes: >
```



```
In [38]: Fare_q1 = data.Fare.quantile(0.25)
         Fare_q3 = data.Fare.quantile(0.75)
         print(Fare_q1)
         print(Fare_q3)
```

```
7.9104
31.0
```

```
In [39]: IQR_Fare=Fare_q3-Fare_q1
         IQR_Fare
```

```
Out[39]: 23.0896
```

In [40]:

```
upperlimit_Fare=Fare_q3+1.5*IQR_Fare
upperlimit_Fare
```

Out[40]:

```
65.6344
```

In [41]:

```
lower_limit_Fare = Fare_q1-1.5*IQR_Fare
lower_limit_Fare
```

Out[41]:

```
-26.724
```

In [42]:

```
median_Fare=data["Fare"].median()
median_Fare
```

Out[42]:

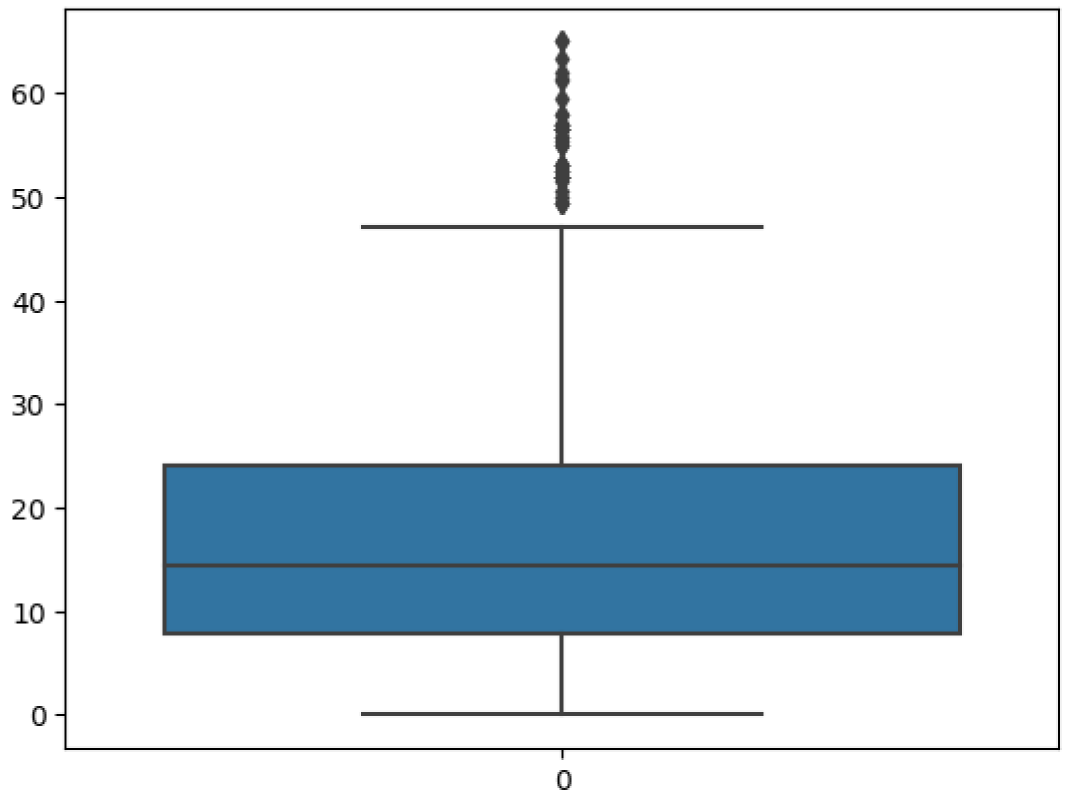
```
14.4542
```

In [43]:

```
data['Fare'] = np.where(
    (data['Fare'] > upperlimit_Fare),
    median_Fare,
    data['Fare']
)
```

```
In [44]: sns.boxplot(data["Fare"])
```

```
Out[44]: <Axes: >
```



```
In [45]: (data["Fare"]>65).sum()
```

```
Out[45]: 0
```

dropping the variables

```
In [46]: data.drop(['Name'],axis=1,inplace=True)
```

In [47]:
```
data
```

Out[47]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	male	22.000000	1	0	A/5 21171	7.2500	G6	S
1	2	1	1	female	38.000000	1	0	PC 17599	14.4542	C85	C
2	3	1	3	female	26.000000	0	0	STON/O2. 3101282	7.9250	G6	S
3	4	1	1	female	35.000000	1	0	113803	53.1000	C123	S
4	5	0	3	male	35.000000	0	0	373450	8.0500	G6	S
...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	male	27.000000	0	0	211536	13.0000	G6	S
887	888	1	1	female	19.000000	0	0	112053	30.0000	B42	S
888	889	0	3	female	29.699118	1	2	W./C. 6607	23.4500	G6	S
889	890	1	1	male	26.000000	0	0	111369	30.0000	C148	C
890	891	0	3	male	32.000000	0	0	370376	7.7500	G6	Q

891 rows × 11 columns

In [48]:
```
data.drop(['Ticket'],axis=1,inplace=True)
```

```
In [49]: data
```

```
Out[49]:
     PassengerId  Survived  Pclass     Sex        Age  SibSp  Parch     Fare  Cabin  Embarked
  0            1         0       3    male  22.000000      1      0   7.2500     G6         S
  1            2         1       1  female  38.000000      1      0  14.4542    C85         C
  2            3         1       3  female  26.000000      0      0   7.9250     G6         S
  3            4         1       1  female  35.000000      1      0  53.1000   C123         S
  4            5         0       3    male  35.000000      0      0   8.0500     G6         S
...          ...       ...     ...     ...        ...    ...    ...      ...    ...       ...
886          887         0       2    male  27.000000      0      0  13.0000     G6         S
887          888         1       1  female  19.000000      0      0  30.0000    B42         S
888          889         0       3  female  29.699118      1      2  23.4500     G6         S
889          890         1       1    male  26.000000      0      0  30.0000   C148         C
890          891         0       3    male  32.000000      0      0   7.7500     G6         Q

891 rows × 10 columns
```

```
In [50]: data.drop(["PassengerId"],axis=1,inplace=True)
In [51]: data
```

```
Out[51]:
     Survived  Pclass     Sex        Age  SibSp  Parch     Fare  Cabin  Embarked
  0         0       3    male  22.000000      1      0   7.2500     G6         S
  1         1       1  female  38.000000      1      0  14.4542    C85         C
  2         1       3  female  26.000000      0      0   7.9250     G6         S
  3         1       1  female  35.000000      1      0  53.1000   C123         S
  4         0       3    male  35.000000      0      0   8.0500     G6         S
...       ...     ...     ...        ...    ...    ...      ...    ...       ...
886         0       2    male  27.000000      0      0  13.0000     G6         S
887         1       1  female  19.000000      0      0  30.0000    B42         S
888         0       3  female  29.699118      1      2  23.4500     G6         S
889         1       1    male  26.000000      0      0  30.0000   C148         C
890         0       3    male  32.000000      0      0   7.7500     G6         Q

891 rows × 9 columns
```

```
In [52]: data.drop(["Cabin"],axis=1,inplace=True)
```

In [53]:
```
data
```

Out[53]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
1	1	1	female	38.000000	1	0	14.4542	C
2	1	3	female	26.000000	0	0	7.9250	S
3	1	1	female	35.000000	1	0	53.1000	S
4	0	3	male	35.000000	0	0	8.0500	S
...	...	...	...	...	...	...	...	...
886	0	2	male	27.000000	0	0	13.0000	S
887	1	1	female	19.000000	0	0	30.0000	S
888	0	3	female	29.699118	1	2	23.4500	S
889	1	1	male	26.000000	0	0	30.0000	C
890	0	3	male	32.000000	0	0	7.7500	Q

891 rows × 8 columns

Splitting the data

In [54]:
```
y=data["Survived"]
```

In [55]:
```
y.head()
```

Out[55]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [56]:

```
data
```

Out[56]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
1	1	1	female	38.000000	1	0	14.4542	C
2	1	3	female	26.000000	0	0	7.9250	S
3	1	1	female	35.000000	1	0	53.1000	S
4	0	3	male	35.000000	0	0	8.0500	S
...	...	...	...	...	...	...	...	...
886	0	2	male	27.000000	0	0	13.0000	S
887	1	1	female	19.000000	0	0	30.0000	S
888	0	3	female	29.699118	1	2	23.4500	S
889	1	1	male	26.000000	0	0	30.0000	C
890	0	3	male	32.000000	0	0	7.7500	Q

891 rows × 8 columns

Encoding

In [57]:

```
from sklearn.preprocessing import LabelEncoder
```

In [58]:

```
le=LabelEncoder()
```

In [59]:

```
data["Sex"]=le.fit_transform(data["Sex"])
```

In [60]:

```
data["Sex"]
```

Out[60]:

```
0      1
1      0
2      0
3      0
4      1
      ..
886    1
887    0
888    0
889    1
890    1
Name: Sex, Length: 891, dtype: int32
```

In [61]:

```
data.head()
```

Out[61]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	1	22.0	1	0	7.2500	S
1	1	1	0	38.0	1	0	14.4542	C
2	1	3	0	26.0	0	0	7.9250	S
3	1	1	0	35.0	1	0	53.1000	S
4	0	3	1	35.0	0	0	8.0500	S

In [62]:

```
data["Embarked"]=le.fit_transform(data["Embarked"])
```

In [63]:
```
data.head()
```

Out[63]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	1	22.0	1	0	7.2500	2
1	1	1	0	38.0	1	0	14.4542	0
2	1	3	0	26.0	0	0	7.9250	2
3	1	1	0	35.0	1	0	53.1000	2
4	0	3	1	35.0	0	0	8.0500	2

In [64]:
```
data["Pclass"].nunique()
```

Out[64]:
```
3
```

In [65]:
```
data["Pclass"].unique()
```

Out[65]:
```
array([3, 1, 2], dtype=int64)
```

In [66]:
```
data["Sex"].unique()
```

Out[66]:
```
array([1, 0])
```

In [67]:
```
data["Embarked"].unique()
```

Out[67]:
```
array([2, 0, 1])
```

Spliting the train and test data

In [68]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(data,y,test_size=0.3,random_state=0)
```

In [69]:
```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[69]:
```
((623, 8), (268, 8), (623,), (268,))
```

Feature Scaling

```
In [70]: from sklearn.preprocessing import StandardScaler
```

```
In [71]: sc=StandardScaler()
```

```
In [72]: x_train=sc.fit_transform(x_train)
```

```
In [73]: x_train
```

```
Out[73]: array([[ 1.25474307, -1.5325562 ,  0.72592065, ..., -0.47299765,
                  0.67925137,  0.56710989],
                [ 1.25474307, -1.5325562 , -1.37756104, ..., -0.47299765,
                 -0.26059483, -2.03075381],
                [-0.79697591,  0.84844757,  0.72592065, ...,  1.93253327,
                  2.26045064,  0.56710989],
                ...,
                [-0.79697591,  0.84844757,  0.72592065, ..., -0.47299765,
                 -0.78281017, -0.73182196],
                [ 1.25474307,  0.84844757, -1.37756104, ..., -0.47299765,
                 -0.03170555,  0.56710989],
                [-0.79697591, -0.34205431,  0.72592065, ...,  0.72976781,
                  1.64661898,  0.56710989]])
```

```
In [74]: x_test=sc.fit_transform(x_test)
```

In [75]:

```
x_test
```

Out[75]:

```
array([[-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.15813988, -1.76531134],
       [-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.72165412,  0.63014911],
       [-0.77151675,  0.77963055,  0.76537495, ...,  0.87064484,
         1.03823178, -0.56758111],
       ...,
       [-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.15847431, -1.76531134],
       [ 1.29614814,  0.77963055, -1.30654916, ..., -0.47809977,
        -0.72607524,  0.63014911],
       [-0.77151675, -1.64991582,  0.76537495, ..., -0.47809977,
         0.92369033, -1.76531134]])
```

In [ ]:

In [ ]: